# Introducing Subjective Knowledge Graphs

Francisco J. Navarrete
*Junta de Andalucía, Spain*
franciscoj.navarrete@juntadeandalucia.es

Antonio Vallecillo
*ITIS Software. Universidad de Málaga, Spain*
av@uma.es

***This is NOT the final version of the paper. The final version will be published on the IEEE website***

***Abstract*—Knowledge-based applications that deal with uncertainty usually represent it by means of a confidence score that expresses the probability that a given fact is true. However, different users may have distinct opinions about the same fact, something that is not considered in existing proposals. This is critical in a number of areas where individual opinions need to be taken into account when making informed decisions, particularly when these are to be made by consensus. This paper introduces *Subjective Knowledge Graphs* (SKG), an extension to Probabilistic Knowledge Graphs that considers the individual opinions of separate users about the same facts, and allows reasoning about them. We show how SKGs can be implemented using standard graph databases and how the results of the queries can be enriched with the associated degrees of uncertainty.**

***Index Terms*—Knowledge Graphs, Uncertainty, Subjective Logic, Opinions, Consensus.**

## I. Introduction

Knowledge Graphs (KGs) [1] provide structured representations of real-world entities and relations in the form of subject-predicate-object (SPO) triples that describe relational facts, e.g., $\langle$*Madrid*, *capitalOf*, *Spain*$\rangle$. KGs are very useful for capturing knowledge from heterogeneous sources, developing question-answering systems, and providing reasoning support for achieving graph completion, inferring new relationships, detecting inconsistencies, or performing triple classification [2].

Generally, knowledge is not free from uncertainty and therefore *Probabilistic* KGs (PKGs) such as ProBase [3], ConceptNet [4], NELL [5], or KV [6] associate every relation fact with a *confidence score*, a weight that represents the likelihood of the relation fact being true or the intensity of a relation [7]. Thus, triples in PKGs are endowed with a fourth component that represents their confidence score, e.g., $\langle$*Mary*, *likes*, *John*, $0.65\rangle$.

Traditionally, degrees of confidence have been modeled using numbers between 0 and 1 that represent probabilities, and reasoning about confidence has been accomplished using probability theory [8], [9]. However, this approach has some limitations, especially when it comes to representing opinions for which users cannot easily express their uncertainty, i.e., their ignorance about the facts they are considering, or their inability to assign an accurate probability to a fact. This type of uncertainty is typically called *second-order probability* or *second-order uncertainty* in statistics and economics, and needs to be explicitly represented, propagated, and taken into account in order to make informed decisions [10]. Besides, different users may hold different opinions about the same fact. These individual opinions need to be reconciled somehow if a decision needs to be made with consensus.

This paper proposes an extension of PKGs to account for subjective opinions of external users about the truthfulness of relation facts, using Subjective Logic [10].

Subjective logic is an extension to Probabilistic and Kleene logics, which explicitly represents uncertainty. Opinions in subjective logic are tuples $w_F^X = (b, d, u, a)$ that express the degree of *belief* ($b$), *disbelief* ($d$) and *uncertainty* ($u$) that a user $X$ holds about a fact $F$. Of course, $b + d + u = 1$. The base rate $a$ represents the prior probability associated to $F$. Usually, $a$ represents the "objective" probability of $F$ as computed using statistical methods (e.g., it corresponds to the current confidence score of PKGs), while $(b, d, u)$ represents the subjective opinion that qualifies the prior probability $a$ according to the user's subjective estimations.

Incorporating subjective opinions into KGs is essential in application domains where information is vague, the sources of information are unreliable, facts can be interpreted differently by separate users, and decisions need to be taken by consensus. Justice is one such domain, as evidences are sometimes vague (and even contradictory), laws admit different interpretations, and reasoning about any legal case must take into account all these uncertainties [11]. Other decision-making systems that fall into this category include, e.g., medical diagnostics applications, autonomous vehicle fleet control systems, digital humanities applications, or stock market investment systems.

Our proposal provides a natural extension to PKGs to account for subjective opinions on the KG relations, which we have called *Subjective* KGs (SKG). This extension is backwards compatible with PKGs, and we show how SKGs can be implemented using standard graph databases and how we can reason about the information contained in them. Some applications are used to demonstrate the proposal, and to show its main advantages and limitations.

The structure of this document is as follows. After this introduction, Sect. II briefly describes the background of our work and also references similar existing proposals. Next, Sect. III presents the example used to motivate our proposal, and describes our approach together with the prototype we have developed as a proof-of-concept for SKGs using Neo4j. Then, Sect. IV presents the results of some evaluation exercises we have conducted to assess our proposal, and discusses its main pros and cons. Finally, Sect. V concludes with an outline of future work.

## II. Background and related work

### A. Knowledge Graphs

Knowledge Graphs (KGs) [1] are labeled directed graphs $(E, R, L)$, where the nodes of $E$ represent entities, the edges of $R$ represent relations between the entities, and the labels of $L$ specify the type of the relation. Thus, elements of a KG can be simply regarded as triples $\langle s, p, o\rangle$ — the so-called *subject-predicate-object* (SPO) triples — where $s, o \in E$, $(s, o) \in R$ and $p \in L$. Such triples provide structured representations of real-world entities and relations that describe relational facts, e.g., $\langle$*Madrid*, *capitalOf*, *Spain*$\rangle$.

KGs enable capturing knowledge from heterogeneous sources, representing it in the form of SPO triples, and reasoning about the information stored in the KG. For example, KG triples can be directly represented as binary facts of first-order logic, which evaluate to true or false, e.g., *capitalOf(Madrid,Spain)*. This encoding allows the use of first-order logic methods and tools to reason about the knowledge described in the KG, such as inferring properties and relations, or developing question-answering systems.

KG information can be incorporated into Machine Learning (ML) tools and techniques using *embedding models* [12]–[15]. These models encode entities as low-dimensional vectors and relations as algebraic operations between them. They accurately capture the similarity of entities and preserve the structure of KGs in the embedding space, and have proved very useful for many knowledge-driven activities, such as graph completion, link prediction, entity discovery, relation extraction, inconsistency detection, or KG triples classification [16].

A KG normally uses a Graph database [17] to store the data, and a reasoning layer to interpret and manipulate it. Graph databases are databases that use graph structures to perform semantic queries; they store the information as *nodes*, *edges*, and *properties*, which makes them fully suitable for storing KGs. Examples of Graph databases include Neo4j, ArangoDB, OrientDB, DGraph, GraphDB, Tigergraph, or JanusGraph, to name a few. They all count with specialized query languages such as Gremlin, Cypher, SPARQL, or GraphQL to interrogate the Graph database and manipulate its information.

### B. Probabilistic Knowledge Graphs

Uncertainty is inherent in many forms of knowledge, so it cannot be neglected in KGs. For example, explicit representation of uncertainty is necessary when modeling random processes (such as chemical reactions or task durations), or when dealing with vague information (e.g., the location of an ancient settlement). Similarly, text comprehension often involves the interpretation of real-world concepts that are ambiguous or inherently vague.

Traditionally, uncertainty in KGs has been considered by assigning each triple a weight, the so-called *confidence score*. Thus, *Probabilistic* KGs (PKGs) such as ProBase [3], ConceptNet [4], NELL [5], or KV [6] associate every relation fact with a *confidence score* that represents the likelihood of the relation fact to be true or the relationship intensity [7]. In these PKGs, triples are endowed with a fourth component that represents such a weight, e.g., $\langle$*university*, *synonym*, *institute*, $0.86\rangle$ or $\langle$*LeeHarveyOswald*, *assassinated*, *JFK*, $0.7\rangle$. In other words, facts are no longer interpreted in Boolean logic, but as probabilities [9], and probability formulas replace truth tables.[1]

The different PKG engines compute the probability associated to each triple in different ways. For example, ConceptNet [4] calculates the confidence score of each triple based on the number and reliability of the sources. NELL [5] uses the maximum expectation algorithm and semi-supervised learning to automatically compute the confidence score of each triple, which can also change over time. In Probase [3], confidence scores provide the prior probability distribution of each concept behind the terms, to support text understanding tasks involving disambiguation. FactChecker [18] is a language-aware approach to infer *believability* of fact candidates, making use of linguistic features to detect if a given source objectively states facts or is speculative and opinionated. In [19], confidence scores are calculated by estimating the truth of the sources from where the information is extracted (in this case, web pages), while the work [20] extracts a KG from noisy and incomplete sources, and therefore the need for specifying probabilities. Other proposals interpret the confidence score as a measure of the relationship intensity, e.g., the relation between an article and a research topic [7].

Reasoning about the information stored in PKGs is supported by different approaches, using probabilistic logic or Datalog languages [21]. Besides, various embedding models for PKGs exist, such as UKGE [16], UOKGE [22] or SUKE [23], which allow accomplishing many knowledge-driven tasks with uncertain triples using ML techniques.

### C. Subjective Logic

Traditionally, uncertainty related to the degree of confidence, likelihood, or belief, has been modeled using numbers between 0 and 1 that represent probabilities, and reasoning about them has been done using probability theory [8], [9].

This approach has some limitations, especially for representing subjective opinions of users who cannot easily express their degree of uncertainty, i.e., they are unable to accurately assign a probability to a fact. In general, forcing users to set probabilities to express their opinions could lead to unreliable conclusions [24], [25]. For instance, when asked about the truthfulness of a fact $F$ for which a user has complete ignorance, it is not the same to assign a probability of 0.5 to it, which would mean that $F$ and $\neg F$ are equally likely (something already informative), than to say "I don't know."

Thus, Subjective logic [10] extends probabilistic logic by explicitly expressing the degree of uncertainty that the expert has about the stated fact.

Let $F$ be a Boolean predicate stating a fact. Let $X$ be a user or any entity able to express opinions about facts — also called a *Belief Agent*. A binomial *opinion* by user $X$ about the truth of $F$ is defined as a quadruple $w_F^X = (b, d, u, a)$ where:

[1] Some PKGs, such as ConceptNet, assign weigths that are not in the range [0,1] but normalization can be easily applied in these cases.

- $b$ (*belief*) is the degree of belief that $F$ is true.
- $d$ (*disbelief*) is the degree of belief that $F$ is false.
- $u$ (*uncertainty*) is the degree of uncertainty about $F$, i.e., the amount of uncommitted belief.
- $a$ (*base rate*) is the prior probability of $F$ without any previous evidence.

These values satisfy the constraints that $b + d + u = 1$, and $b, d, u, a \in [0, 1]$. Intuitively, the *base rate* of an opinion represents the *objective* probability that can be assigned to the fact using *a priori* evidences or statistical estimates, whilst the other elements of the tuple represent the *subjective* degrees of belief, disbelief and uncertainty about the fact assigned by the expert. Thus, regardless of the value of the prior probability, different users can express their subjective opinions about the same fact, including their individual degrees of uncertainty.

The *projected probability* (or *projection*) of an opinion is defined as $P(w_F^X) = b + au$. The projected probability permits combining the objective and subjective values into one single probability, which modifies the prior base rate according to the user opinion. Kleene logic does not include base rates and therefore probability projections cannot be derived.

For example, two separate users can have different opinions about the same fact, e.g., the assassination of JKF. Thus, they can use the previous confidence score $0.7$ as base rate of their subjective opinions, and express them as $(0.1, 0.4, 0.5, 0.7)$ and $(0.8, 0.0, 0.2, 0.7)$. The projections of these two opinions are $0.45$ and $0.94$, with respective uncertainties of $0.5$ and $0.2$. This allows representing in a more faithful manner the individual views of the two users about the same fact.

In addition to the common logical operators (`and`, `or`, `implies`, etc.) used to combine the opinions of the same expert about different facts, Subjective logic implements *fusion* operators for combining the opinions of different users about the same fact [10]. The goal is to produce a single opinion that better reflects the collection of opinions, or is closer to the truth than each opinion in isolation. This is key for allowing collaborative modeling and enabling cooperative work between users when they need to reach consensus decisions.

To represent and operate with Subjective logic values, in [25] we defined a primitive datatype, `SBoolean`, that extends type `Boolean` with uncertainty information. A `SBoolean` value is defined by the quadruple $(b, d, u, a)$ that represents the opinion in Subjective logic.

A probability $p$ can be *lifted* to a subjective opinion in type `SBoolean` by the quadruple $(p, 1 - p, 0, p)$. Similarly, a subjective opinion $w$ can be *projected* to a probability by its projection $P(w) = b + au$. This embedding provides a proper subtyping relation between these two types, and when $u = 0$, we get the standard probability values and logic.

Finally, a property that will be later used by our proposal is that any opinion $w_F^X = (b, d, u, a)$ can also be expressed as $w_F^X = (p, u, a)$, where $p$ is the opinion projection. To show this, recall that $p = b + au$, so knowing $(b, d, u, a)$ we can obtain $(p, u, a)$. Similarly, if we know $(p, u, a)$, then $b = p - au$ and $d = 1 - b - u$. This is important because we realized that for most users it is easier to think in terms of prior probability $(a)$, posterior probability $(p)$, and degree of uncertainty $(u)$, than in terms of degrees of belief, disbelief and uncertainty. In all cases, $p \geq au$ because $b + d + u = 1$ should hold.

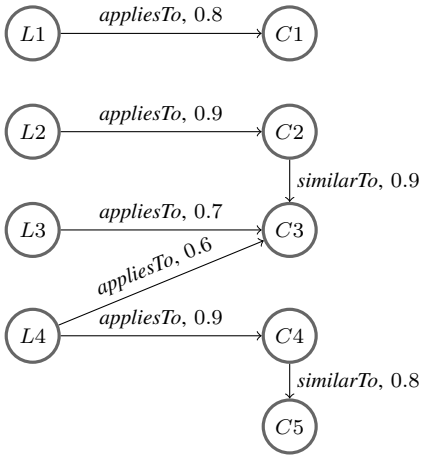


Figure 1. A PKG with Laws $(L)$ that can be applied to legal cases $(C)$.

For example, instead of users $X$ and $Y$ providing the opinions $(0.1, 0.4, 0.5, 0.7)$ and $(0.8, 0.0, 0.2, 0.7)$ for the previous uncertain triple $\langle$*LeeHarveyOswald*, *assassinated*, *JFK*, $0.7\rangle$, it would be enough for them to express their opinions as $(0.45, 0.5)$ and $(0.94, 0.2)$, respectively.

Subjective logic has been used in several domains, such as preventive maintenance [26], self-adaptive cyber-physical systems [27], or decision support systems [28], [29], and also in the Semantic Web domain for ontology alignment [30], [31], annotation evaluation [32], [33] and inconsistency handling [34], [35]. In the context of KGs, subjective logic was proposed to weigh semantic similarities between concepts in WordNet [36], [37]. Beyond that, to our knowledge this is the first time it is applied to KGs and implemented using a commercial database.

## III. Proposal

### A. Motivating example

To illustrate our proposal and motivate the problem it addresses, we will use an example application in the field of justice. Any allegedly criminal act may be regulated by different laws that may be applicable and may carry different penalties. When a legal case reaches the court, it must be decided to whom the case is assigned. The objective is to identify the law that should be applied in the first instance, in order to assign the case to the appropriate prosecutor.

Suppose that five criminal acts $(C1, .., C4)$ are brought to court, and we identify four laws that can be applied to them $(L1, .., L4)$. Predicate *appliesTo* models such a relationship. Besides, some cases may be *similar* to others and then the same law could be applied to them. We can represent the following situation using the following facts, which are accompanied by a confidence score that specifies their likelihood of being true. They are graphically shown in the PKG of Fig. 1.

$\langle\, L1, \textit{appliesTo}, C1, 0.8 \,\rangle$ $\langle\, L2, \textit{appliesTo}, C2, 0.9 \,\rangle$
$\langle\, L3, \textit{appliesTo}, C3, 0.7 \,\rangle$ $\langle\, L4, \textit{appliesTo}, C3, 0.6 \,\rangle$
$\langle\, L4, \textit{appliesTo}, C4, 0.9 \,\rangle$ $\langle\, C2, \textit{similarTo}, C3, 0.9 \,\rangle$
$\langle\, C4, \textit{similarTo}, C5, 0.8 \,\rangle$

The degree of application of a law to a case is calculated using the history of previous case resolutions. The similarity between two cases can be computed using several methods, for example, by calculating the distance in the Euclidean space whose dimensions represent the properties and characteristics of the legal case, or by applying ML techniques such as those used to measure the similarity of commercial products [38].

Using PKG deductive reasoning, it is easy to infer from these triples the most probable law to apply to each case:

| Case | $C1$ | $C2$ | $C3$ | $C4$ | $C5$ |
|---|---|---|---|---|---|
| Law | $L1$ | $L2$ | $L2$ | $L4$ | $L4$ |
| Likelihood | 0.8 | 0.9 | 0.81 | 0.9 | 0.72 |

The problem is that different legal experts may have different opinions about the reliability of the algorithms used to compute the confidence scores. For example, judge John may not be sure that cases $C2$ and $C3$ are so similar, or that the law $L3$ really applies to $C3$. This would result in individual *views* on the PKG, each tailored according to the opinions of the individual jurists. How to represent these individual views is the target of SKGs, and will be the focus of the next subsection.

### B. Subjective KGs

A Subjective Knowledge Graph (SKG) is a directed labeled graph whose elements are of the form $\langle s, p, o, a, W \rangle$, where:

- $s$ and $o$ are nodes of the graph, connected by the edge $(s, o)$, which is labeled as $p$; i.e. $\langle s, p, o \rangle$ constitutes a SPO-triple as in a traditional KG.
- $a$ is the confidence score assigned to the triple $\langle s, p, o \rangle$; i.e. $\langle s, p, o, a \rangle$ constitutes a PKG element.
- $W = \{(X_1; p_1, u_1), (X_2; p_2, u_2), ..., (X_n; p_n, u_n)\}$ is a list of tuples, where $X_1, .., X_n$ are the identifiers of nodes representing users (*belief agents*) that hold opinions about the triple $\langle s, p, o \rangle$; $p_1, .., p_n$ are the users' posterior opinions corresponding to the original base rate $a$; and $u_1, .., u_n$ represent their associated uncertainties.

The list $W$ represents the opinions of individual users about the fact. An important property of $W$ is that a belief agent can appear at most once in the list; that is, users can only have one opinion per fact. If a belief agent does not appear in the list, it means that he/she does not have any particular opinion about the fact, and therefore agrees with the confidence score initially assigned to it. Thus, a PKG can be considered a SKG whose $W$ lists are all empty, just as a basic KG can be considered as a PKG whose confidence scores are all equal to 1.

For example, the following tuple represents the fact previously mentioned that states that Lee Harvey Oswald assassinated JFK with a probability of 0.7, together with the beliefs of the two users, $X$ and $Y$, who are now able to express their individual opinions about this fact in the SKG:

$$\langle\, \textit{LeeHarveyOswald}, \textit{assassinated}, \textit{JFK}, 0.7, \{(X; 0.45, 0.5),\ (Y; 0.94, 0.2)\} \,\rangle$$

### C. Reasoning with SKG tuples

Going back to the legal system example described in Sect. III-A, suppose that Mary, John and Lucy are three jurists who do not fully agree with the confidence scores assigned to some of the facts expressed in the previous PKG. Now they can explicitly express their uncertainties as follows:

$\langle\, L1, \textit{appliesTo}, C1, 0.8, \{(\textit{Lucy}; 0.6, 0.3)\} \,\rangle$
$\langle\, L2, \textit{appliesTo}, C2, 0.9, \{(\textit{Mary}; 0.5, 0.5),\ (\textit{John}; 0.9, 0.1)\} \,\rangle$
$\langle\, L3, \textit{appliesTo}, C3, 0.7, \{(\textit{Mary}; 0.6, 0.5),\ (\textit{Lucy}; 0.9, 0.2),\ (\textit{John}; 0.8, 0.1)\} \,\rangle$
$\langle\, L4, \textit{appliesTo}, C3, 0.6, \{(\textit{Mary}; 0.9, 0.2),\ (\textit{Lucy}; 0.7, 0.2),\ (\textit{John}; 0.4, 0.6)\} \,\rangle$
$\langle\, L4, \textit{appliesTo}, C4, 0.9, \{(\textit{Mary}; 0.8, 0.5),\ (\textit{John}; 0.7, 0.1)\} \,\rangle$
$\langle\, C2, \textit{similarTo}, C3, 0.9, \{(\textit{Lucy}; 0.7, 0.3),\ (\textit{John}; 0.9, 0.1)\} \,\rangle$
$\langle\, C4, \textit{similarTo}, C5, 0.8, \{(\textit{Mary}; 0.4, 0.5),\ (\textit{Lucy}; 0.5, 0.5),\ (\textit{John}; 0.9, 0.1)\} \,\rangle$

Using these subjective opinions, four decision tables can be obtained. They are shown in Table I below. The first one (*Initial PKG*) does not take into account any subjective opinions. It is the same as the table shown in Sect. III-A, but is included here to facilitate comparison with the other three tables. The individual tables contain not only information about the posterior probability assigned by the corresponding belief agent, but also its degree of uncertainty. This is very informative because decisions based on opinions with a high uncertainty should be avoided [10]. Shaded cells indicate a significant change in the law to apply with respect to the initial decision (i.e., those shown in the *Initial PKG* table), or opinions with low probabilities or high uncertainty ($\geq 0.5$).

We can see how John's opinions do not produce any substantial change in the decisions. In contrast, law $L4$ should apply to case $C3$ according to Mary, and $L3$ according to Lucy. In addition, there are laws that perhaps should not be applied because their associated uncertainty is high. All these computations have been performed using the same operations that were used for PKGs, although now on `SBoolean` values [25].

### D. Fusing opinions

A distinctive feature of Subjective logic is that it defines a set of operations to combine the opinions of separate users about the same fact, with the goal of reaching consensus

Table I
DECISION TABLES FOR THE APPLICATION OF LAWS TO LEGAL CASES.

*Initial PKG*

| Case | $C1$ | $C2$ | $C3$ | $C4$ | $C5$ |
|---|---|---|---|---|---|
| Law | $L1$ | $L2$ | $L2$ | $L4$ | $L4$ |
| Likelihood | 0.8 | 0.9 | 0.81 | 0.9 | 0.72 |

*John*

| Case | $C1$ | $C2$ | $C3$ | $C4$ | $C5$ |
|---|---|---|---|---|---|
| Law | $L1$ | $L2$ | $L2$ | $L4$ | $L4$ |
| Prob./Unc. | $0.8, 0.0$ | $0.9, 0.1$ | $0.81, 0.1$ | $0.7, 0.1$ | $0.63, 0.1$ |

*Mary*

| Case | $C1$ | $C2$ | $C3$ | $C4$ | $C5$ |
|---|---|---|---|---|---|
| Law | $L1$ | $L2$ | $L4$ | $L4$ | $L4$ |
| Prob./Unc. | $0.8, 0.0$ | $0.5, 0.5$ | $0.9, 0.2$ | $0.8, 0.5$ | $0.32, 0.4$ |

*Lucy*

| Case | $C1$ | $C2$ | $C3$ | $C4$ | $C5$ |
|---|---|---|---|---|---|
| Law | $L1$ | $L2$ | $L3$ | $L4$ | $L4$ |
| Prob./Unc. | $0.6, 0.3$ | $0.9, 0.0$ | $0.9, 0.2$ | $0.9, 0.0$ | $0.45, 0.3$ |

decisions [10]. Each operator is more suitable in a particular situation, although in this case all agents observe the same situation at the same time, and all opinions should be taken into account, even when they are very uncertain. Therefore, the **Averaging Belief Fusion (ABF)** operator is recommended [39].

We can use this operator to decide the law that needs to be applied to case $C3$, because the three jurists disagreed on which was the most appropriate. The following table shows the subjective opinions of the three legal experts about the applicability of laws $L2$, $L3$ and $L4$ to case $C3$. The last two rows show the fused opinions of each column, and the projection of the fused opinion, respectively. Figures are rounded to one decimal digit in `SBoolean` values. Shaded cells indicate the best option in each case.

| | $L2$ | $L3$ | $L4$ |
|---|---|---|---|
| *John* | $(0.7, 0.2, 0.1, 0.8)$ | $(0.7, 0.2, 0.1, 0.7)$ | $(0.1, 0.3, 0.6, 0.6)$ |
| *Mary* | $(0.3, 0.5, 0.2, 0.8)$ | $(0.3, 0.2, 0.5, 0.7)$ | $(0.7, 0.0, 0.3, 0.6)$ |
| *Lucy* | $(0.5, 0.3, 0.2, 0.8)$ | $(0.8, 0.0, 0.2, 0.7)$ | $(0.6, 0.2, 0.2, 0.6)$ |
| **ABF** | $(0.6, 0.3, 0.1, 0.8)$ | $(0.7, 0.1, 0.2, 0.7)$ | $(0.5, 0.2, 0.3, 0.6)$ |
| Proj. | 0.68 | 0.81 | 0.71 |

Since the degrees of uncertainty are reasonable, the three jurists agree to apply the $L3$ law to the $C3$ case.

### *E. Implementing SKGs in Graph databases*

To evaluate the proposed approach we have implemented our Subjective KG examples in a graph database, namely Neo4j [17]. A graph database is an online database management system with create, read, update and delete (CRUD) methods that expose a graph data model. Graph databases are generally built for use with transactional systems. They are typically optimized for transactional performance (in order to be used for dealing with huge sets of data) and designed with transactional integrity and operational availability in mind.

As previously mentioned, graph database consists of *nodes*, *relationships*, *properties* and *labels*.

- Nodes contain properties, which are represented in terms of arbitrary key-value pairs. In Neo4j, keys are of type String and values of Java primitive datatypes, as well as arrays of these types.
- Nodes can be tagged with one or more labels. Labels group nodes together and indicate the roles they play within the dataset.
- Relationships connect the nodes and structure the graph. A relation always has an address, a unique name and a source and target node: there are no dangling relations.
- Relations in a graph naturally form *paths*. Querying or traversing the graph involves following paths. Because of the fundamentally path-oriented nature of the data model, most path-based graph database operations are closely aligned with the way the data is laid out, making them extremely efficient.

Figure 2 shows the legal system example described in Sect. III-A represented in Neo4j (see also Fig. 1). The query that builds such as graph is shown at the top of Fig. 2, expressed in the Cypher language:

```
match (n)-[:appliesTo|similarTo]-(o)
return *
```

The query simply returns all nodes that are connected through any of the *appliesTo* or *similarTo* relationships, along with the corresponding relations.

As mentioned above, subjective knowledge graphs are probabilistic knowledge graphs that incorporate the subjective opinions of users about the relationships established between entities, complementing the objective probability of the relational fact with the expression of the degree of confidence and uncertainty assigned to it by the corresponding belief agents.

To represent the subjective opinions of agents and adapting to the structural characteristics of Neo4j, we have defined, for each existing relationship between two entities (nodes): $(i)$ a property called "`probability`" that stores the confidence score of the fact and $(ii)$ four arrays, which store the identifiers of the belief agents, and their degrees of belief, disbelief and uncertainty, respectively. The confidence score of the record, serves as the common base rate of all the subjective opinions defined in the arrays. Although we initially thought of using the more compact representation of opinions using the pairs $(p, u)$, we realized that internally the original representation $(a, b, u)$ was more efficient for performing operations, both logical (e.g., `and`) or for fusing opinions (e.g., `averagingBeliefFusion`).

Queries can be performed on SKGs exactly in the same way as in normal KGs. The difference now is that the results are accompanied by both the confidence score (i.e., "`probability`") and by the subjective opinions of those belief agents that expressed their opinions on the results. E.g., returning to the example in the previous section, we can issue a query for knowing whether law $L4$ can be applied to the legal case $C3$ as follows:

```
match (:L4)-[:appliesTo]-(:C3)
return *
```

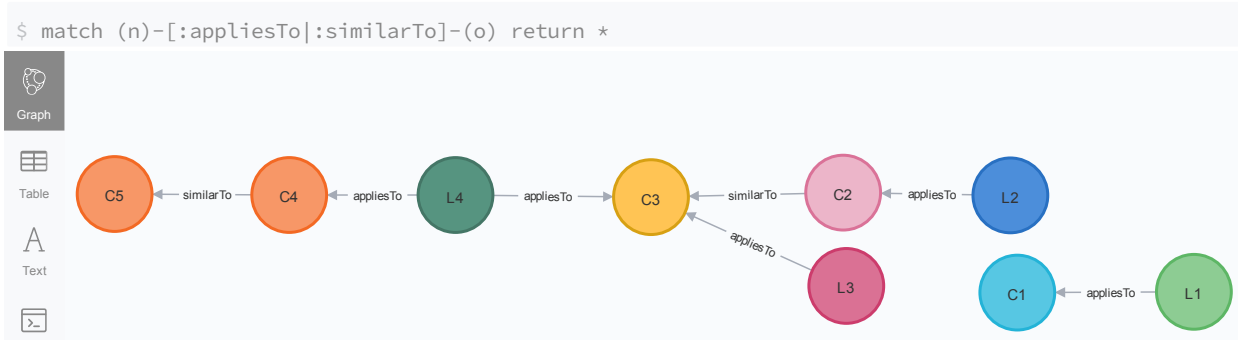


Figure 2. Neo4j graph of the legal system example.

The result obtained by Neo4j is shown in Fig. 3. We can see that there is a relation, whose confidence score is 0.6, and that three belief agents, *Lucy*, *John* and *Mary*, have expressed some opinions on this: $(Lucy; 0.58, 0.22, 0.2)$, $(John; 0.04, 0.36, 0.6)$ and $(Mary; 0.7, 0.0, 0.3)$.

| "r" |
| --- |
| {"uncertainty":[0.2,0.6,0.3],"belief":[0.58,0.04,0.7],"probability":0.6,"p_opinion":["Lucy","John","Mary"],"disbelief":[0.22,0.36,0.0]} |

Figure 3. Result of the query about the relation between $L4$ and $C3$.

In addition, we have developed a Neo4j plugin that incorporates the `SBoolean` Java library [25], [40] to implement that datatype and all its related operations. The plugin also implements some specific functions for performing the basic operations on opinions as represented in the graph database.

First, method `sboolean.util.foldSBooleanAnd` builds the set of combined opinions of several agents over the relationships of a path, using `and` operations. The precise definition of such an operation is as follows:

```
sboolean.util.foldSBooleanAnd(
  relations :: LIST? OF RELATIONSHIP?,
  agents :: LIST? OF STRING?
) :: (STRING?)
```

Method `sboolean.util.foldAverageBeliefFusion` calculates the average belief fusion of the relations given in a path, over a list of opinion agents, as follows.

```
sboolean.util.foldAverageBeliefFusion(
  relations :: LIST? OF RELATIONSHIP?,
  agents :: LIST? OF STRING?
) :: (relation::LIST? OF RELATIONSHIP?, opinion::STRING?)
```

Method `sboolean.util.addSBooleanOpinion` adds the subjective opinion of an agent to the knowledge base. The opinion is specified in terms of its projection and uncertainty.

```
sboolean.util.addSBooleanOpinion(
   relation :: RELATIONSHIP?, agent :: STRING?,
   opinion_projection :: FLOAT?, uncertainty :: FLOAT?
) :: (relation :: RELATIONSHIP?, opinion :: STRING?)
```

Using these operations, we can easily build updates on the SKG. For example, the following two operations incorporate the opinions of belief agents Lucy and Mary on relations $\langle L1, appliesTo, C1, r\rangle$ and $\langle L2, appliesTo, C2, r\rangle$, respectively.

```
match (:L1)-[r:appliesTo]-(:C1)
call sboolean.util.addSBooleanOpinion(r,'Lucy',0.6,0.3)
yield relation, opinion
return relation, opinion
```

```
match (:L2)-[r:appliesTo]-(:C2)
call sboolean.util.addSBooleanOpinion(r,'Mary',0.5,0.5)
yield relation, opinion
return relation, opinion
```

Obtaining the applicable laws for the legal case $C3$ considering the opinions of the three jurists *John*, *Mary* and *Lucy*, as well as their fused opinions, can be simply written as follows:

```
match p=(:Law)-[*1..2]-(:C3)
call sboolean.util.foldAverageBeliefFusion(
       relationships(p),
       ['John','Mary','Lucy']
   )
yield relation, opinion
return p as path, opinion
order by opinion desc
```

The resulting knowledge graph is shown in Fig. 4, which determines that law $L2$ could be applied to case $C2$, and laws $L3$ and $L4$ to case $C3$.

Figure 5 shows the quantitative data detailing the opinions on the resulting paths, as well as the averaging belief fusion of these opinions. The first tuple indicates the best positioned law to apply to the $C3$ case, which corresponds to the $L3$ law, as previously mentioned.

We could also be interested in automatically computing the individual opinions of each belief agent about every path of the query, as we showed in the table displayed in Sect. III-D. For this, we simply calculate the paths and then compute the opinion of each agent about each path by combining his/her opinions on each relation of the path using the `foldSBooleanAnd` operator. This query can be simply written as follows:

```
match p=(n:Law)-[*1..2]-(:C3)
call sboolean.util.foldAverageBeliefFusion(
       relationships(p),
       ['John','Mary','Lucy'] )
yield relation, opinion
return p as path, opinion,
       sboolean.util.foldSBooleanAnd(
           relationships(p),
           ['John','Mary','Lucy']
       ) as foldSBooleanAnd
order by n.name
```

The results of the query are shown in Fig. 6. The first column shows the complete paths for each

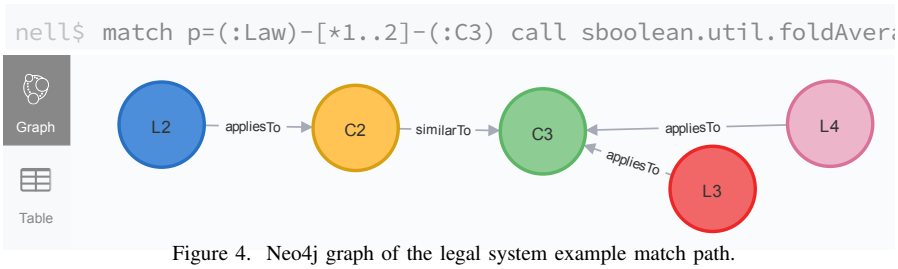


Figure 4. Neo4j graph of the legal system example match path.

| "path" | "opinion" |
|---|---|
| [{"name":"L3"},{"uncertainty":[0.5,0.2,0.1],"belief":[0.25,0.76,0.7300000000000001],"probability":0.7,"p_opinion":["Mary","Lucy","John"],"disbelief":[0.25,0.03999999999999998,0.16999999999999999]},{"name":"C3"}] | "{projection: 0.805883, sboolean: (0.682353,0.141176,0.176471,0.7)}" |
| [{"name":"L4"},{"uncertainty":[0.2,0.6,0.3],"belief":[0.58,0.04,0.7],"probability":0.6,"p_opinion":["Lucy","John","Mary"],"disbelief":[0.22,0.36,0.0]},{"name":"C3"}] | "{projection: 0.71, sboolean: (0.53,0.17,0.3,0.6)}" |
| [{"name":"L2"},{"uncertainty":[0.5,0.1],"belief":[0.04999999999999999,0.7100000000000001],"probability":0.9,"p_opinion":["Mary","John"],"disbelief":[0.44999999999999996,0.18999999999999992]},{"name":"C2"},{"name":"C2"},{"uncertainty":[0.3,0.1],"belief":[0.42999999999999994,0.81],"probability":0.9,"p_opinion":["Lucy","John"],"disbelief":[0.2700000000000001,0.08999999999999994]},{"name":"C3"}] | "{projection: 0.640729, sboolean: (0.532103,0.333791,0.134106,0.81)}" |

Figure 5. The consensus decisions ordered.

triple $\langle L2, appliesTo, C3\rangle$, $\langle L3, appliesTo, C3\rangle$, and $\langle L4, appliesTo, C3\rangle$. The second one shows the resulting combined consensus opinions for each path. The third column displays the individual opinions of each jurist about each fact.

All the artefacts related to this example, including the graph database with the data, the queries and the plugin we have developed for supporting subjective logic operations in Neo4j, are available from [41].

## IV. Initial Evaluation

To evaluate our proposal we have performed an initial set of experiments, which are described in this section. They are not intended to provide an exhaustive evaluation, but rather to assess the main non-functional features of our approach and to discuss its pros and cons.

### A. Performance and scalability

In this section we are interested in investigating two main issues. The first one is whether our proposal is able to deal with large Knowledge Graphs, such as those used by other existing PKGs. Second, we would like to know the performance penalty introduced when subjective opinions are incorporated to the KG, in terms of response time of the queries.

#### A.1. Querying a large PKG

To respond to the first question, we generated a PKG in Neo4j using data obtained from the NELL project. NELL (Never-Ending Language Learner) [5] is a knowledge base of structured information that mirrors the content of the Web. The main objective of NELL is to build a never-ending machine learning system that acquires the ability to extract structured information from unstructured web pages. The inputs to NELL include an initial ontology defining hundreds of categories (e.g., *person*, *sportsTeam*, *fruit*, *emotion*) and relations (e.g., *playsOnTeam*(*athlete*,*sportsTeam*), *playsInstrument*(*musician*,*instrument*)) that NELL is expected to extract, and 10 to 15 seed examples of each category and relation.

Using the Neo4j importing facilities, we created a Neo4j database with the 2,121,179 nodes and 644,899 relations currently available from the NELL PKG [42], incorporating their corresponding confidence scores. Then, we performed several queries, each one trying to highlight a different characteristic of the KG that we wanted to analyze.

All execution times shown here were taken using the `profile` function defined in Neo4j, which can accompany any query to compute its execution time. All experiments were run on a desktop machine with Archlinux version 5.12.6-arch1-1, an i5-5200U CPU, with 2 cores at 2.2 GHz and hyper-threading (4 threads) and 12 GB of RAM. We used Neo4j community edition version 4.2.2 with Java 11 (OpenJDK 11.0.11) configured with the default options except for the heap size which was set to 8 GB. Each query was performed 10 times, discarding the first run (as warm up). We executed the 10 executions together in the same machine, but after each execution we waited until the garbage collector had released the used memory.

**Basketball teams**. With this query we wanted to know, for each basketball player in the database, which is the most probable team the athlete plays for. This is an example of a query that involves two relationships: a person *plays* a sport (in this case, Basketball), and a person *playsIn* a team. The confidence score of each resulting triple is computed by multiplying the probabilities of the corresponding relations.

| "path" | "opinion" | "foldSBooleanAnd" |
|---|---|---|
| [{"name":"L2"},{"uncertainty":[0.5,0.1],"belief":[0.04999999999999999,0.7100000000000001],"probability":0.9,"p_opinion":["Mary","John"],"disbelief":[0.44999999999999996,0.18999999999999992]},{"name":"C2"},{"name":"C2"},{"uncertainty":[0.3,0.1],"belief":[0.42999999999999994,0.81],"probability":0.9,"p_opinion":["Lucy","John"],"disbelief":[0.2700000000000001,0.08999999999999994]},{"name":"C3"}] | "{projection: 0.640729, sboolean: (0.532103,0.333791,0.134106,0.81)}" | "{projection: 0.72, sboolean: (0.6471,0.2629,0.09,0.81, agent: John)}{projection: 0.45, sboolean: (0.258158,0.505,0.236842,0.81, agent: Mary)}{projection: 0.63, sboolean: (0.514895,0.343,0.142105,0.81, agent: Lucy)}" |
| [{"name":"L3"},{"uncertainty":[0.5,0.2,0.1],"belief":[0.25,0.76,0.7300000000000001],"probability":0.7,"p_opinion":["Mary","Lucy","John"],"disbelief":[0.25,0.03999999999999998,0.16999999999999999]},{"name":"C3"}] | "{projection: 0.805883, sboolean: (0.682353,0.141176,0.176471,0.7)}" | "{projection: 0.8, sboolean: (0.73,0.17,0.1,0.7, agent: John)}{projection: 0.6, sboolean: (0.25,0.25,0.5,0.7, agent: Mary)}{projection: 0.9, sboolean: (0.76,0.04,0.2,0.7, agent: Lucy)}" |
| [{"name":"L4"},{"uncertainty":[0.2,0.6,0.3],"belief":[0.58,0.04,0.7],"probability":0.6,"p_opinion":["Lucy","John","Mary"],"disbelief":[0.22,0.36,0.0]},{"name":"C3"}] | "{projection: 0.71, sboolean: (0.53,0.17,0.3,0.6)}" | "{projection: 0.4, sboolean: (0.04,0.36,0.6,0.6, agent: John)}{projection: 0.88, sboolean: (0.7,0.0,0.3,0.6, agent: Mary)}{projection: 0.7, sboolean: (0.58,0.22,0.2,0.6, agent: Lucy)}" |

Figure 6. Subjective opinions of the three legal experts about the applicability of laws $L2$, $L3$ and $L4$ to case $C3$.

```
profile match p=(n:basketball)-[r0:concept_players]-(o)-
                        [r1:concept_athleteplaysforteam]-(g)
return o.bestLiteralName as player,
       g.bestLiteralName as team,
reduce (acc = 1, r IN relationships(p) |
            acc * r.probability) as likelihood, p
order by likelihood desc
```

The result of the query yielded 45 players and took an average time of 7.11 ms.

**Hotels to follow the league**. With this query we wanted to know, for each player of any discipline of the most likely team he plays for, the team stadium and the possible hotels where to stay to see them play live. This query involves navigating paths of 5 relationships and is expected to return a high number of results. Again, the confidence score is calculated by multiplying the probabilities of the corresponding relations.

```
profile match p=(n)-[r1:concept_players]-(o)-
       [r2:concept_athleteplaysforteam]-(g)-
       [r3:concept_stadiumhometeam]-(s)-
       [r4:concept_citystadiums]-(c)-
       [r5:concept_cityhotels]-(a)
return o.bestLiteralName as player,
       g.bestLiteralName as team,
       a.bestLiteralName as hotel,
reduce (acc = 1, r IN relationships(p) |
            acc * r.probability) as likelihood, p
order by likelihood desc
```

The result of the query yielded 4,291 results and took an average time of 27.89 ms.

**Cities and their relations**. This query requests information about all cities included in the knowldge graph, and computes the probabilities associated with the nodes that can be reached from each city through a direct relation. In contrast to the previous query, which used a long navigation path, this query focuses on many direct relations.

```
profile match p=(g:city)-[r0:IsGeneralization]-(c)-[r1]-(o)
where type(r1)<>"IsNamedLiteral"and
      type(r1)<>"IsGeneralization"
return c.bestLiteralName as entity1,
       o.bestLiteralName as entity2,
reduce (acc = 1, r IN relationships(p) |
            acc * r.probability) as likelihood,p
order by likelihood desc
```

The result of the query yielded 35,953 results and took an average time of 45.44 ms.

### *A.2. Incorporating opinions to the PKG*

The second exercise consisted in analyzing the performance penalty caused by the introduction of subjective opinions in the PKG.

For this we populated the Neo4j database with opinions from different belief agents and run the queries again, this time asking the system to compute both the individual opinions of the selected belief agents on the results, and also their fused opinions. The values of the opinions of the different belief agents were randomly assigned.

**Basketball teams with opinions**. We added the (invented) opinions of three renowned NBA basketball specialists and commentators: *Kevin Harlan*, *Gus Jonson* and *Marv Albert*. The original query can be reformulated as follows, in order to consider the opinions of the three experts.

```
profile match p=(n:basketball)-[r0:concept_players]-(o)-
                        [r1:concept_athleteplaysforteam]-(g)
call sboolean.util.foldAverageBeliefFusion(
       relationships(p),
       ['Kevin Harlan','Gus Jonsone','Marv Albert'])
yield relation, opinion
return o.bestLiteralName as player,
       g.bestLiteralName as team,
       opinion as likelihood, p
order by likelihood desc
```

The result of the query yielded again the same 45 players, but in a few cases the corresponding teams were different when the opinions of the experts were taken into account, as expected. The execution time of this query was 11.55, which represents a ratio of 1.6 with respect to the previous one.

**Hotels to follow the league with opinions**. To perform the queries we used the above three belief agents (*Kevin Harlan*, *Gus Jonson* and *Marv Albert*), whose opinions were added to all triples in the database. Again, the values of all these opinions were randomly chosen. Using these opinions, the query can be written as follows.

```
profile match p=(n)-[r1:concept_players]-(o)-
  [r2:concept_athleteplaysforteam]-(g)-
  [r3:concept_stadiumhometeam]-(s)-
  [r4:concept_citystadiums]-(c)-[r5:concept_cityhotels]-(a)
call sboolean.util.foldAverageBeliefFusion(
       relationships(p),
       ['Kevin Harlan','Gus Jonsone','Marv Albert'])
yield relation, opinion
return o.bestLiteralName as player,
       g.bestLiteralName as team,
       a.bestLiteralName as hotel,
       opinion as likelihood, p
order by likelihood desc
```

The result of the query yielded again 4291 hotels. The execution time of this query was 43.89 ms, which represents a ratio of 1.57 with respect to the previous one.

**Cities and their relations with opinions**. To perform the third query we used two belief agents (*PN* and *AV*), whose opinions were added to all triples in the database. Again, the values of all these opinions were randomly chosen. Using these opinions, the query can be written as follows.

```
profile match p=(g:city)-[r0:IsGeneralization]-(c)-[r1]-(o)
where type(r1)<>"IsNamedLiteral" and
      type(r1)<>"IsGeneralization"
call sboolean.util.foldAverageBeliefFusion(
      relationships(p), ['AV','PN'])
yield relation, opinion
return c.bestLiteralName as entity1,
       o.bestLiteralName as entity2,
       opinion as likelihood,p
order by likelihood desc
```

The result of the query yielded 35,953 relations. The execution time of this query was 58.56 ms, which represents a ratio of 1.28 with respect to the previous one.

In summary, the performance penalty caused by the introduction of subjective opinions and its evaluation in the queries ranged between 1.28 and 1.6, which seems to be acceptable when times are in the order of a few milliseconds.

### B. Methodology

As mentioned in the introduction, our proposal is intended to be used in domains where expert opinions (i.e., second-order probabilities) must be considered to make informed decisions, beyond the probabilities automatically inferred applications. Examples include, for example, the diagnoses of a medical team or the opinions of several jurists about a legal case.

Thus, we foresee our proposal used in such a way that subjective opinions by experts are progressively added to the KG, enriching the knowledge base. Every time an expert is asked about his/her opinion, it is recorded so that it can be used in later consultations to the knowledge base.

Of course, not all queries must take into account expert opinions. One of the advantages of our proposal is that it provides a natural extension of the PKG. This means that those users who do not want to take subjective opinions into account will be able to query the PKG exactly as they did before.

### C. Discussion

This section briefly discusses the main advantages and limitations that we have identified during the development of our prototype. First, we have been able to successfully implement our proposal using a commercial database such as Neo4j, which demonstrates its feasibility. The response times obtained are reasonable, although we think there is plenty of room for improvement because our design was rather direct and was not optimized for performance. In particular, the limitations of Neo4j in terms of storing complex data structures complicates the processes of querying, inserting and deleting opinions in the relations, forcing us to maintain four related lists of values to store both the agent and the components of the subjective opinions (belief, disbelief and uncertainty). Future revisions will attempt to simplify the storage model.

Asking domain experts to specify their opinions may represent a significant challenge, because they are not used to quantify their beliefs — think for instance of doctors or judges. The use of literals such as *Probably*, *Possibly*, or *Unlikely* [43] could be of help, but this needs to be empirically evaluated. Moreover, the loss of information caused by this simplification needs to be properly quantified.

We also realized that some facts are only temporarily true, or that opinions have a temporary validity. Some proposals assign a time interval to the triples, indicating their period of validity. For example, the work [44] combines temporal information with confidence scores (expressed in terms of probabilities) in the triples, allowing more expressive knowledge representation. The incorporation of temporal validity in our proposal, both for facts and opinions, is part of our planned work.

Finally, most PKGs currently treat facts as independent random variables. However, in reality, many triples are correlated. The consequences and effects of such correlations on the corresponding opinions should also be further studied.

## V. Conclusions

In this paper we have proposed an extension to Probabilistic Knowledge Graphs that enables the incorporation of domain experts' individual opinions to qualify the prior probability specified in the PKG triples. This second-order probability, together with its associated uncertainty, can be automatically propagated when navigating the relations of a graph to resolve a query. Moreover, individual opinions by different agents on a given fact can be combined, or *fused* in Subjective logic terms, to reach consensus opinions.

We have demonstrated our proposal by implementing it using a commercial database, Neo4j, and developing several knowledge graphs. We showed how the SKG can be queried using the Cypher language, and how the results are now more informative because their uncertainty can be automatically calculated and made explicit. This will allow users to avoid making decisions with too high a degree of uncertainty, something that may entail a high risk.

There are several lines of work that we plan to address next. In particular, we want to tackle the issues identified in Sect. IV, develop further case studies and conduct more evaluation experiments to assess the usability and efficiency of our proposal. This proposal is part of a larger project that aims to provide an automated decision support system for the classification and analysis of court cases using legal

documents, past decisions, and the subjective opinions of judges. Therefore, interesting challenges still lie ahead.

## ACKNOWLEDGMENT

We would like to thank the reviewers for their insightful comments and constructive suggestions. This work was funded by the Spanish Research Project PGC2018-094905-B-100.

## REFERENCES

[1] C. Gutiérrez and J. F. Sequeda, "Knowledge graphs," *Commun. ACM*, vol. 64, no. 3, pp. 96–104, 2021.

[2] S. Ji, S. Pan, E. Cambria, P. Marttinen, and P. S. Yu, "A survey on knowledge graphs: Representation, acquisition and applications," *CoRR*, vol. abs/2002.00388, 2020.

[3] W. Wu, H. Li, H. Wang, and K. Q. Zhu, "Probase: a probabilistic taxonomy for text understanding," in *Proc. of ACM SIGMOD'12*. ACM, 2012, pp. 481–492.

[4] R. Speer, J. Chin, and C. Havasi, "ConceptNet 5.5: An open multilingual graph of general knowledge," in *Proc. of AAAI'17*. AAAI Press, 2017, pp. 4444–4451.

[5] T. M. Mitchell *et al.*, "Never-ending learning," *Commun. ACM*, vol. 61, no. 5, pp. 103–115, 2018.

[6] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, N. Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang, "Knowledge vault: a web-scale approach to probabilistic knowledge fusion," in *Proc. of KDD'14*. ACM, 2014, pp. 601–610.

[7] M. Nayyeri, G. M. Gil, S. Vahdati, F. Osborne, A. Kravchenko, S. Angioni, A. Salatino, D. R. Recupero, E. Motta, and J. Lehmann, "Link prediction using numerical weights for knowledge graph completion within the scholarly domain," in *Proc. of ESWC'21*, ser. LNCS. Springer, 2021.

[8] W. Feller, *An Introduction to Probability Theory and Its Applications*, 3rd ed. Wiley, 1968, vol. 1.

[9] B. de Finetti, *Theory of Probability: A critical introductory treatment*. John Wiley & Sons, 2017.

[10] A. Jøsang, *Subjective Logic – A Formalism for Reasoning Under Uncertainty*, ser. Artificial Intelligence: Foundations, Theory, and Algorithms. Springer, 2016.

[11] A. Jøsang and V. A. Bondi, "Legal reasoning with subjective logic," *Artif. Intell. Law*, vol. 8, no. 4, pp. 289–315, 2000.

[12] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, "Translating embeddings for modeling multi-relational data," in *Proc. of NIPS'13*, ser. NeurIPS Proceedings, vol. 26. Curran Associates, Inc., 2013, pp. 2787–2795.

[13] Z. Wang, J. Zhang, J. Feng, and Z. Chen, "Knowledge graph embedding by translating on hyperplanes," in *Proc. of AAAI'14*. AAAI Press, 2014, pp. 1112–1119.

[14] B. Yang, W. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *Proc. of ICLR'15*, 2015.

[15] T. Trouillon, J. Welbl, S. Riedel, É. Gaussier, and G. Bouchard, "Complex embeddings for simple link prediction," in *Proc. of ICML'16*, ser. JMLR Proceedings, vol. 48. JMLR.org, 2016, pp. 2071–2080.

[16] X. Chen, M. Chen, W. Shi, Y. Sun, and C. Zaniolo, "Embedding uncertain knowledge graphs," in *Proc. of AAAI'19*. AAAI Press, 2019, pp. 3363–3370.

[17] I. Robinson, J. Webber, and E. Eifrem, *Graph Databases: New Opportunities for Connected Data*, 2nd ed. O'Reilly Media, Inc., 2015.

[18] N. Nakashole and T. M. Mitchell, "Language-Aware Truth Assessment of Fact Candidates," in *Proc. of ACL'14*. The Association for Computer Linguistics, 2014, pp. 1009–1019.

[19] X. L. Dong, E. Gabrilovich, K. Murphy, V. Dang, W. Horn, C. Lugaresi, S. Sun, and W. Zhang, "Knowledge-based trust: Estimating the trustworthiness of web sources," *Proc. VLDB Endow.*, vol. 8, no. 9, pp. 938–949, 2015.

[20] G. M. S. Namata Jr. and L. Getoor, "Identifying graphs from noisy and incomplete data," in *Proc. of ACM SIGKDD Workshop on Knowledge Discovery from Uncertain Data*. ACM, 2009, pp. 23–29.

[21] L. Bellomarini, E. Laurenza, E. Sallinger, and E. Sherkhonov, "Reasoning under uncertainty in knowledge graphs," in *Proc. of RuleML+RR'20*, ser. LNCS, vol. 12173. Springer, 2020, pp. 131–139.

[22] K. Boutouhami, J. Zhang, G. Qi, and H. Gao, "Uncertain ontology-aware knowledge graph embeddings," in *Proc. of JIST'19*, ser. CCIS, vol. 1157. Springer, 2019, pp. 129–136.

[23] J. Wang, K. Nie, X. Chen, and J. Lei, "SUKE: embedding model for prediction in uncertain knowledge graph," *IEEE Access*, vol. 9, pp. 3871–3879, 2021.

[24] A. Jøsang, "A logic for uncertain probabilities," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 9, no. 3, pp. 279–212, 2001.

[25] P. Muñoz, L. Burgueño, V. Ortiz, and A. Vallecillo, "Extending OCL with Subjective Logic," *Journal of Object Technology*, vol. 19, no. 3, pp. 3:1–15, Oct. 2020, special Issue dedicated to Martin Gogolla on his 65th Birthday.

[26] B. Uzunoglu, "An adaptive bayesian approach with subjective logic reliability networks for preventive maintenance," *IEEE Trans. Reliab.*, vol. 69, no. 3, pp. 916–924, 2020.

[27] A. Petrovska, S. Quijano, I. Gerostathopoulos, and A. Pretschner, "Knowledge aggregation with subjective logic in multi-agent self-adaptive cyber-physical systems," in *Proc. of SEAMS'20*. ACM, 2020, pp. 149–155.

[28] C. González-Fernández, J. Cabezas, A. Fernández-Isabel, and I. M. de Diego, "Combining multi-agent systems and subjective logic to develop decision support systems," in *Proc. of IPMU'20*, ser. CCIS, vol. 1237. Springer, 2020, pp. 143–157.

[29] F. M. Zennaro and A. Jøsang, "Using subjective logic to estimate uncertainty in multi-armed bandit problems," *CoRR*, vol. abs/2008.07386, 2020. [Online]. Available: https://arxiv.org/abs/2008.07386

[30] D. Hooijmaijers and M. Stumptner, "Improving integration with subjective combining of ontology mappings," in *Proc. of ISMIS'08*, ser. LNCS, vol. 4994. Springer, 2008, pp. 552–562.

[31] K. Juszczyszyn, "A subjective logic-based framework for aligning multiple ontologies," in *Proc. of KES'04*, ser. LNCS, vol. 3214. Springer, 2004, pp. 1194–1200.

[32] D. Ceolin, A. Nottamkandath, and W. J. Fokkink, "Automated evaluation of annotators for museum collections using subjective logic," in *Proc. of IFIPTM'12*, ser. IFIP Advances in Information and Communication Technology, vol. 374. Springer, 2012, pp. 232–239.

[33] A. Nottamkandath, J. Oosterman, D. Ceolin, and W. J. Fokkink, "Automated evaluation of crowdsourced annotations in the cultural heritage domain," in *Proc. of URSW@ISWC'14*, ser. CEUR Workshop Proceedings, vol. 1259. CEUR-WS.org, 2014, pp. 25–36. [Online]. Available: http://ceur-ws.org/Vol-1259/ursw2014_submission_5.pdf

[34] M. Sensoy, J. Z. Pan, A. Fokoue, M. Srivatsa, and F. Meneguzzi, "Using subjective logic to handle uncertainty and conflicts," in *Proc. of TrustCom'12*. IEEE Computer Society, 2012, pp. 1323–1326.

[35] S. Babalou and B. König-Ries, "A subjective logic based approach to handling inconsistencies in ontology merging," in *Proc. of OTM'19*, ser. LNCS, vol. 11877. Springer, 2019, pp. 588–606.

[36] D. Ceolin, A. Nottamkandath, and W. J. Fokkink, "Subjective logic extensions for the semantic web," in *Proc. of URSW'12*, ser. CEUR Workshop Proceedings, vol. 900. CEUR-WS.org, 2012, pp. 27–38. [Online]. Available: http://ceur-ws.org/Vol-900/paper3.pdf

[37] ——, "Bridging gaps between subjective logic and semantic web," in *Proc. of URSW 2011-2013, Revised Selected Papers*, ser. LNCS, vol. 8816. Springer, 2014, pp. 242–264.

[38] A. Ravi, S. Repakula, U. K. Dutta, and M. Parmar, "Buy me that look: An approach for recommending similar fashion products," *CoRR*, vol. abs/2008.11638, 2020. [Online]. Available: https://arxiv.org/abs/2008.11638

[39] A. Jøsang, D. Wang, and J. Zhang, "Multi-source fusion in subjective logic," in *Proc. of FUSION'17*. IEEE, 2017, pp. 1–8.

[40] Atenea Research Group, "Uncertainty in software models - Git repository," https://github.com/atenearesearchgroup/uncertainty, 2021.

[41] F. J. Navarrete and A. Vallecillo, "Subjective Knowledge Graphs. Git repository," 2021. [Online]. Available: https://github.com/atenearesearchgroup/subjectiveKGs

[42] Carnegie Mellon University, "Nell: Never-ending language learning," 2020. [Online]. Available: http://rtw.ml.cmu.edu/rtw/

[43] P. Martín-Rodilla and C. Gonzalez-Perez, "Conceptualization and non-relational implementation of ontological and epistemic vagueness of information in digital humanities," *Informatics*, vol. 6, no. 2, p. 20, 2019.

[44] M. W. Chekol, G. Pirrò, J. Schoenfisch, and H. Stuckenschmidt, "Marrying uncertainty and time in knowledge graphs," in *Proc. of AAAI'17*, S. P. Singh and S. Markovitch, Eds. AAAI Press, 2017, pp. 88–94.